Itanium® 2 Processor Microarchitecture Overview

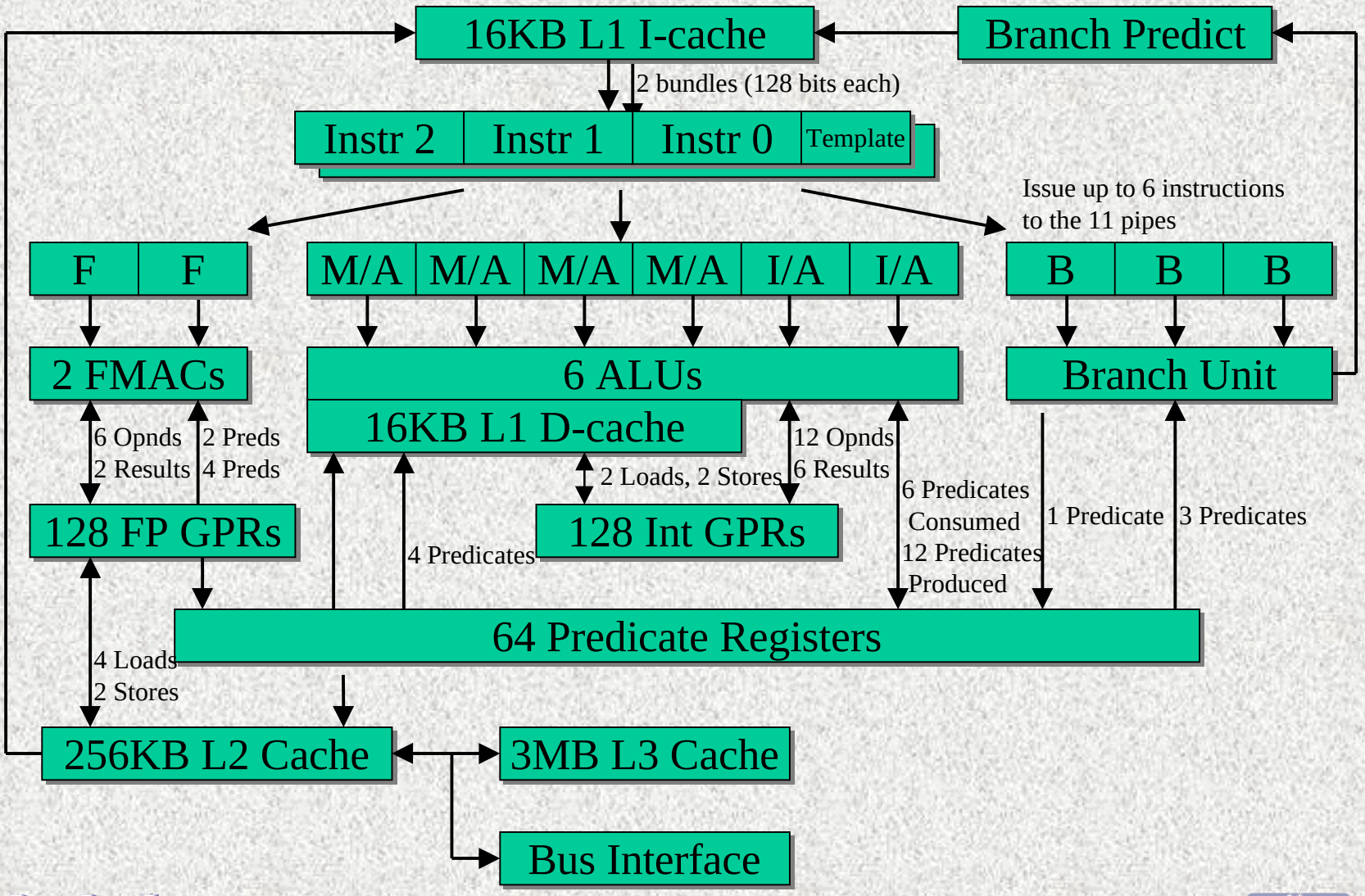
Don Soltis, Mark Gibson

Cameron McNairy

Hot Chips 14, August 2002

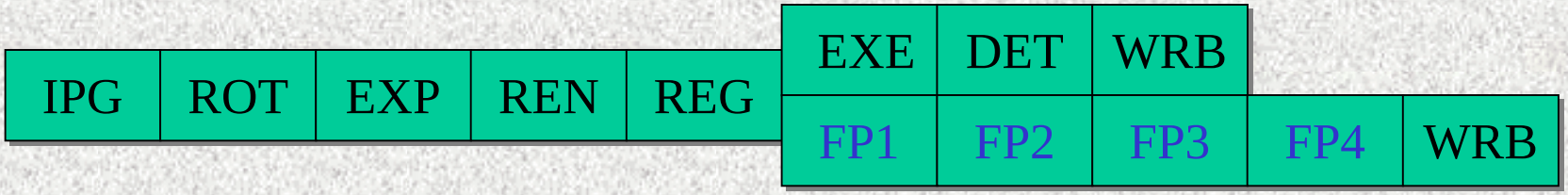
Itanium® 2 Processor Overview

Block Diagram



Itanium® 2 Processor Overview

Main Execution Unit Pipeline



IPG: Instruction Pointer Generate, Instruction address to L1 I-cache
ROT: Present 2 Instruction Bundles from L1 I-cache to dispersal hardware
EXP: Disperse up to 6 instruction syllables from the 2 instruction bundles
REN: Rename (or convert) virtual register IDs to physical register IDs
REG: Register file read, or bypass results in flight as operands

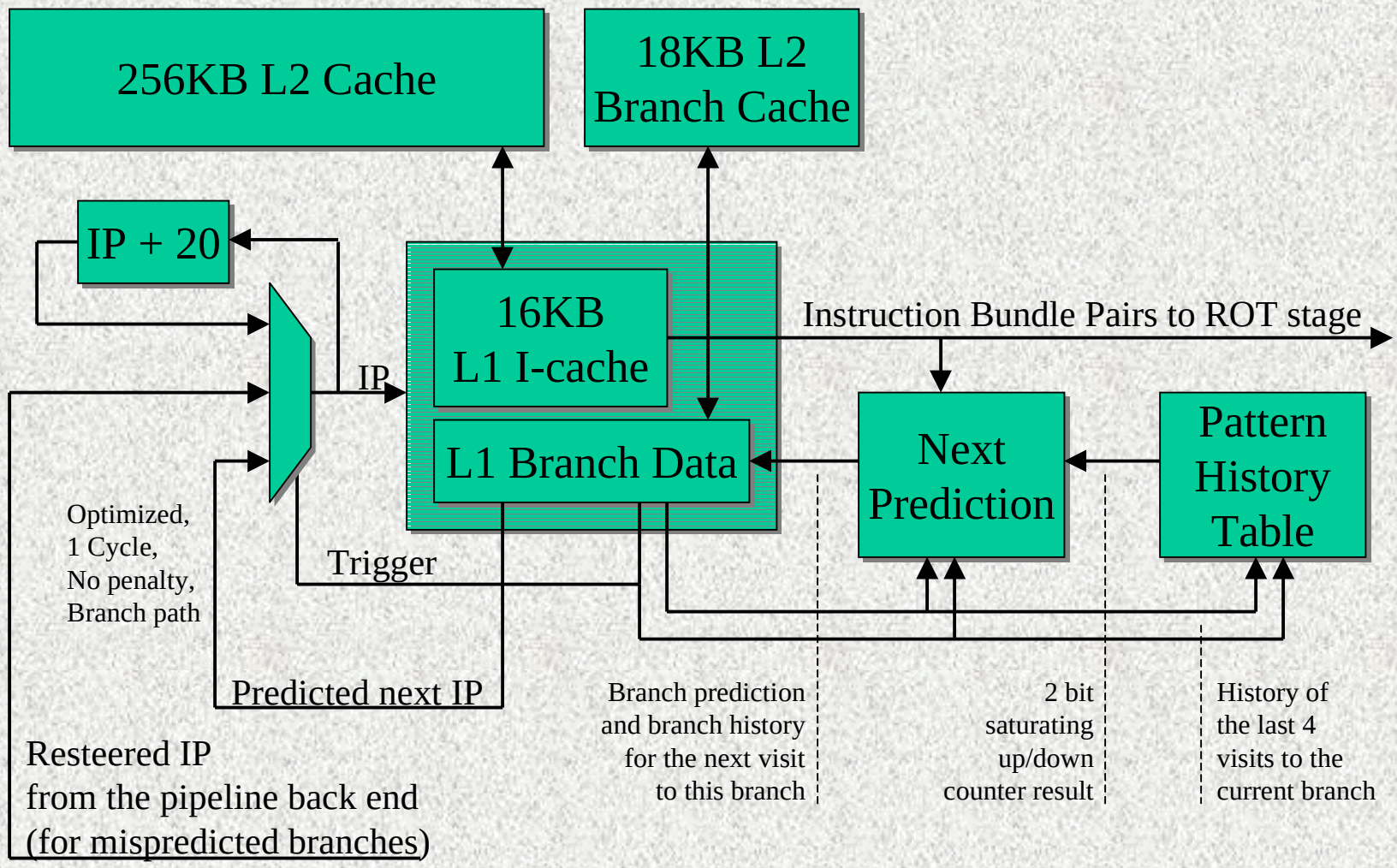
EXE: Execute integer instructions; generate results and predicates
DET: Detect exceptions, traps, etc.

FP1-4: Execute floating point instructions; generate results and predicates

WRB: Write back results to the register file (architectural state update)

Itanium® 2 Processor Overview

Branch Prediction



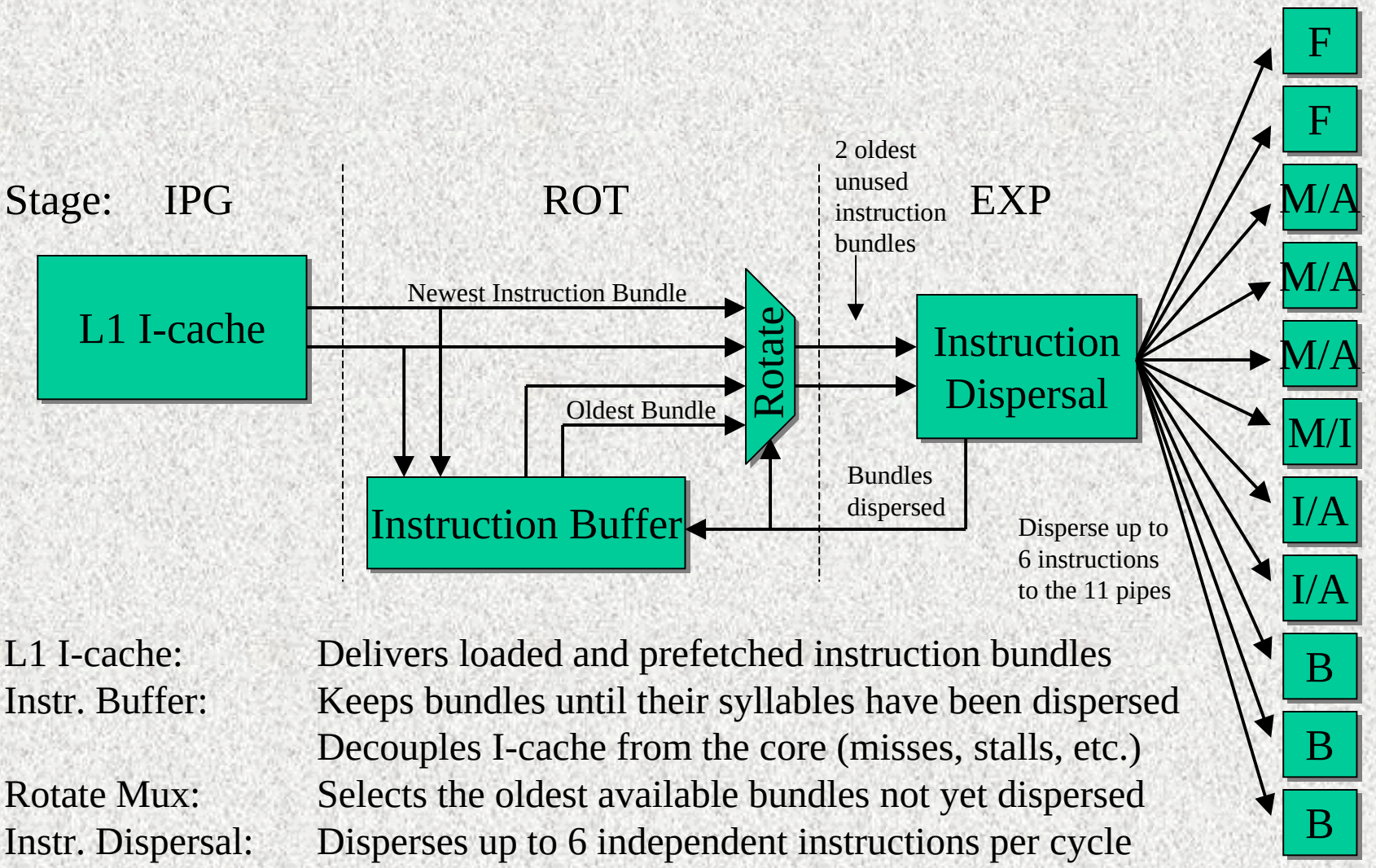
Itanium® 2 Processor Overview

Branch Prediction

- Optimized for single cycle, no penalty branch prediction
  - L1 Branch Data is kept in the L1 I-cache for fast access and low overhead
  - Branch target is stored with history in L1 to save address computation time
  - L1 Branch Data fills from L2 Branch cache and writes through to L2 Branch cache like other cache data
- 2 levels of Branch Data: L1 I-cache and L2 Branch cache
  - L2 Branch Data keeps track of 8-12K branch bundles (or up to 36K branches)
  - Optimized for good branch prediction over many branches; provides better performance for large programs
- L1 Branch Data is encoded to allow for a variable number of branches per instruction bundle pair.
  - 64 bit Branch Data entries cover one instruction bundle pair
  - Fewer branches per bundle pair can store more information and provide better branch prediction (optimized for 2 branches per bundle)
- Can resolve 3 branches per cycle.

Itanium® 2 Processor Overview

Pipeline Front End and Dispersal



L1 I-cache: Delivers loaded and prefetched instruction bundles

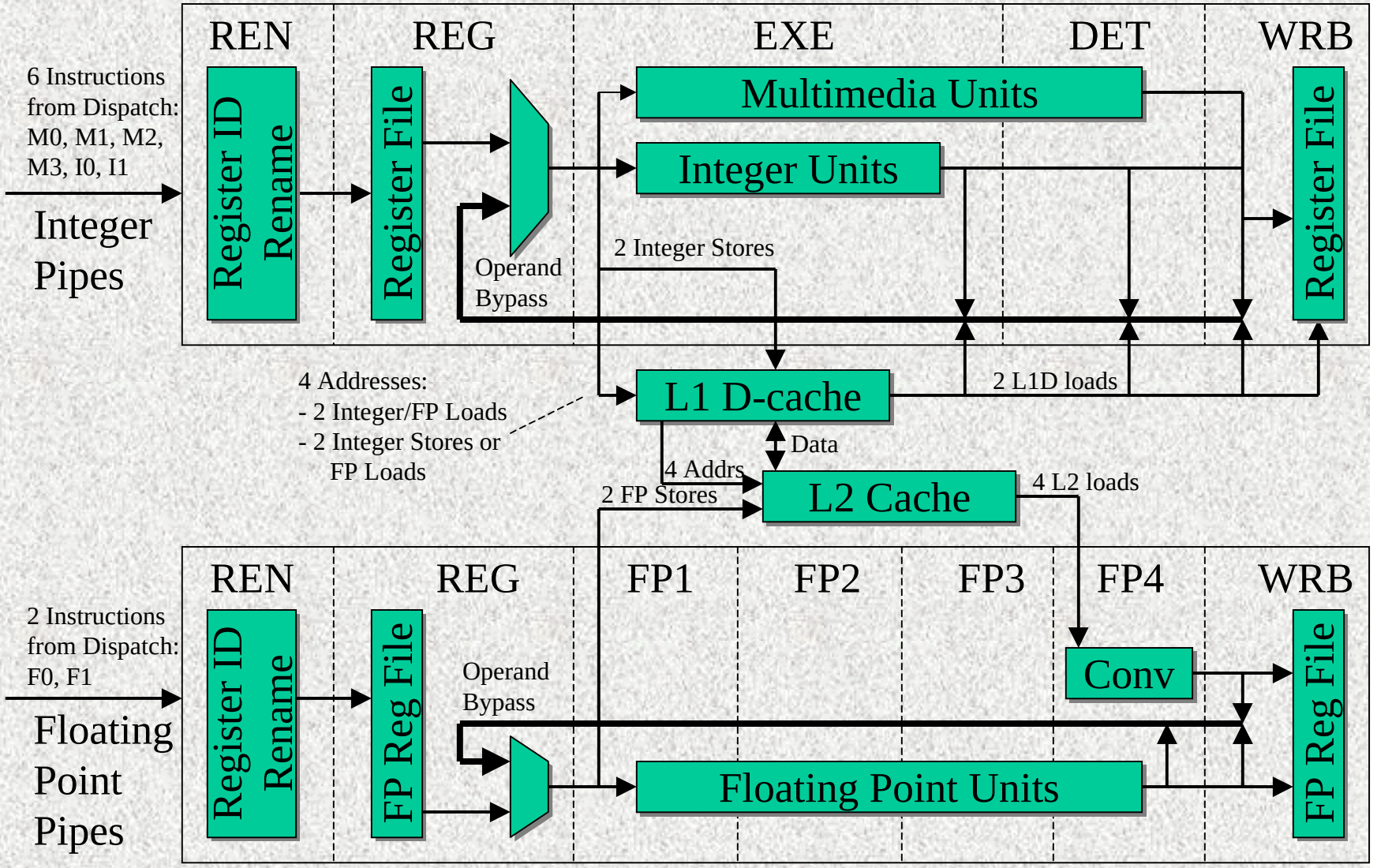
Instr. Buffer: Keeps bundles until their syllables have been dispersed
Decouples I-cache from the core (misses, stalls, etc.)

Rotate Mux: Selects the oldest available bundles not yet dispersed

Instr. Dispersal: Disperses up to 6 independent instructions per cycle

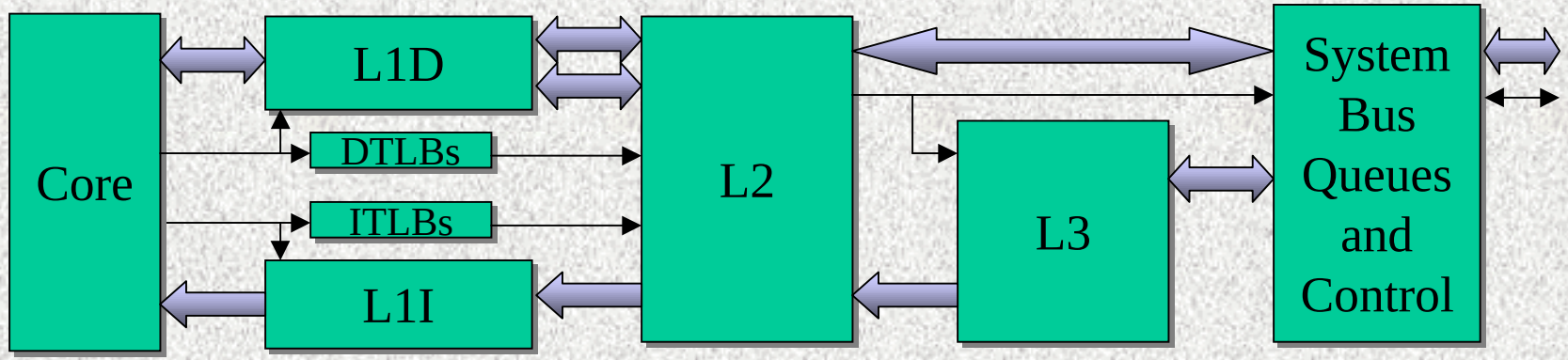
Itanium® 2 Processor Overview

Execution Unit Pipelines



Itanium® 2 Processor Overview

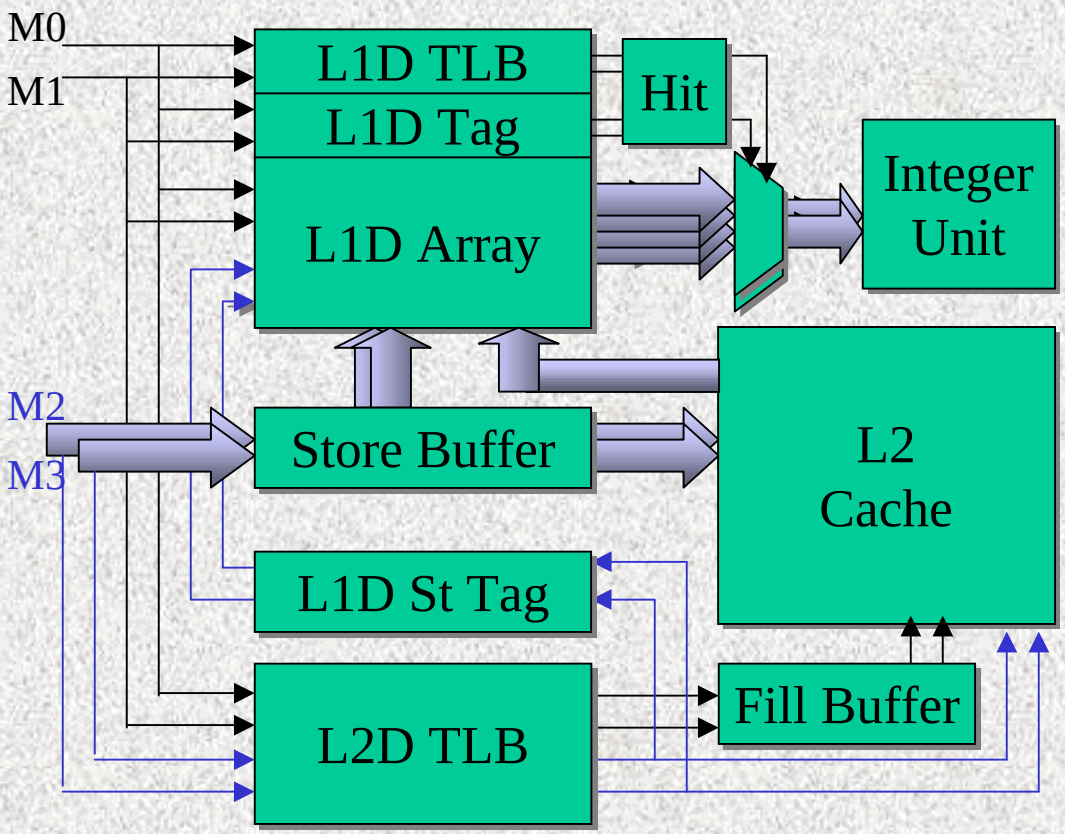
Memory Subsystem



	L1I	L1D	L2	L3
Size	16KB	16KB	256KB	3MB
Line Size	64B	64B	128B	128B
Ways	4	4	8	12
Replacement	LRU	NRU	NRU	NRU
Latency	1	1	5+, 6+, 7+	12+
Bandwidth	R: 32 B/c	R: 16 B/c W: 16 B/c	R: 48 B/c W: 32 B/c	R: 32 B/c W: 32 B/c
Ports	T: 2 D: 1	T: 2+2 D: 2+	T: 4 D: 4	T: 1 D: 1

Itanium® 2 Processor Overview

L1D cache



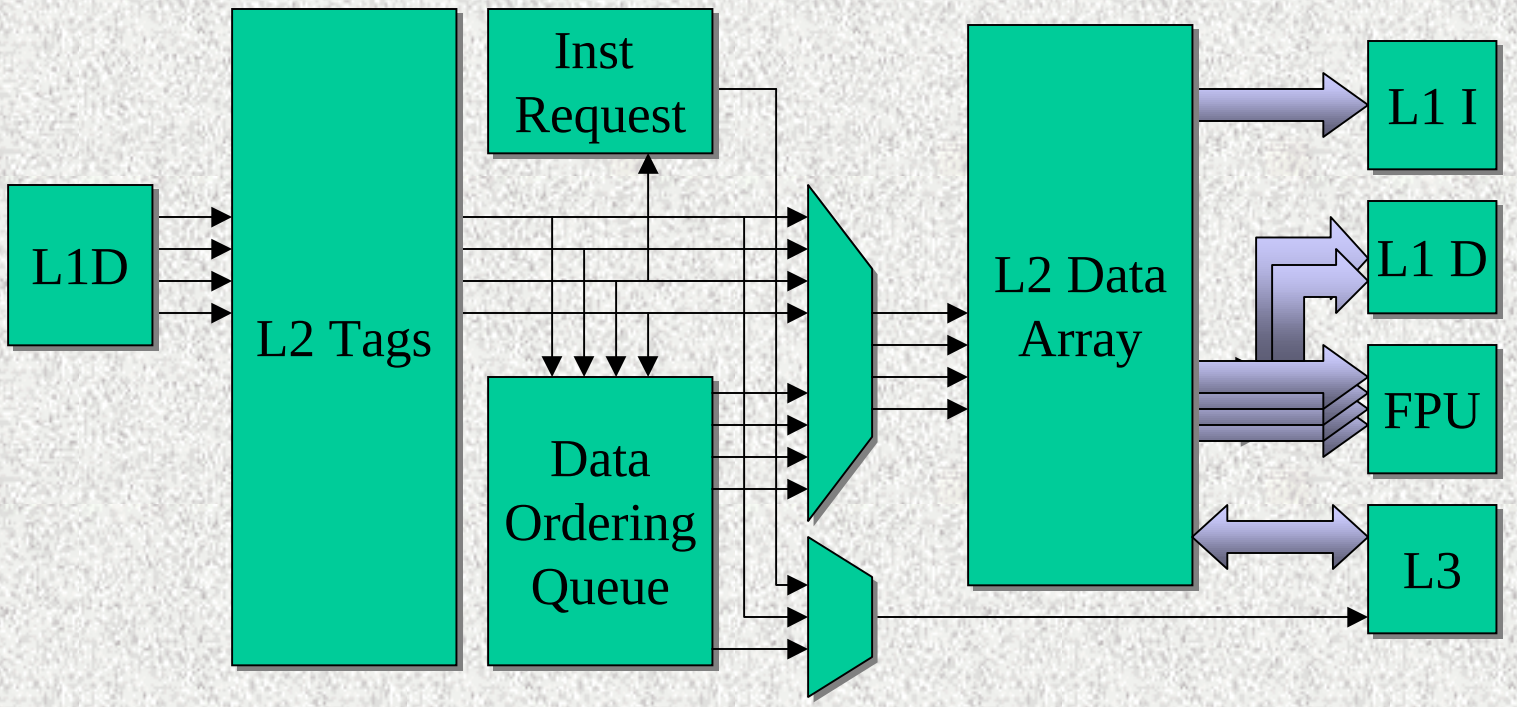
- Integer data only
- 1 cycle latency
- 2 loads + 2 stores/cy
- Load path prevalidated
- Store path conventional
- In order execution
- Non-blocking, 8 fills
- Write-through stores

- L1D TLB supports 4K pages
- 32 entry fully associative
- Used only for loads for prevalidation

- L2D TLB supports 4K to 4G pages
- 128 entry fully associative
- Accessed in parallel with L1D TLB

Itanium® 2 Processor Overview

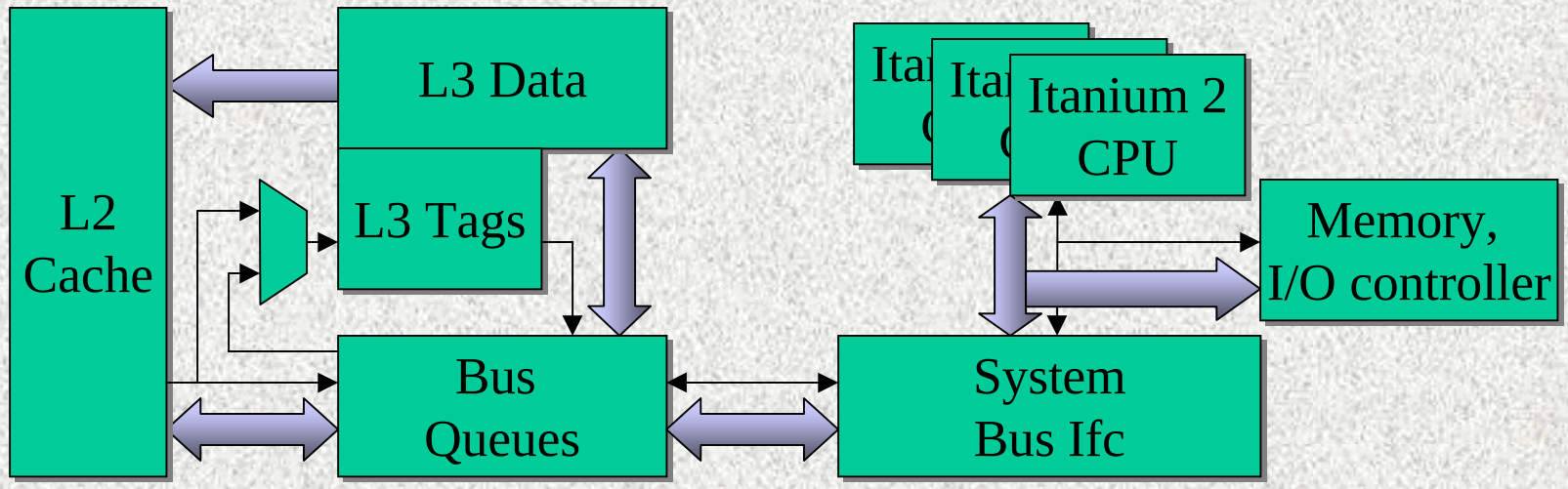
L2 out-of-order cache



- Unified cache
  - 8 entry inst queue
  - 32 entry data queue
- 4 memory ops per cycle
- 32 entry request queue
- 16 banks, 16B per bank
- Out-of-order execution
- Non-blocking, 16 fills

Itanium® 2 Processor Overview

L3 cache and Bus Interface



- 3mb, 12 way unified cache (I+D)
- On chip cache
- 12, 16 cycle latency (integer)
- Tag – 1 access per cycle
- Load data – 1 line per 4 cycles
- Write data – 1 line per 4 cycles
- Out-of-order execution
- Non-blocking, 16 miss buffer (BRQ)

- Split bus – independent address and data busses
- Address & Control Bus with 50 bits of physical addressing
- 128 bits wide (16 Bytes) data bus with 400 MT/s (double pumped 200MHz bus) for 6.4 Gb/s
- Support up to 19 outstanding bus transactions per agent
- Glueless 4 CPU MP on one FSB)

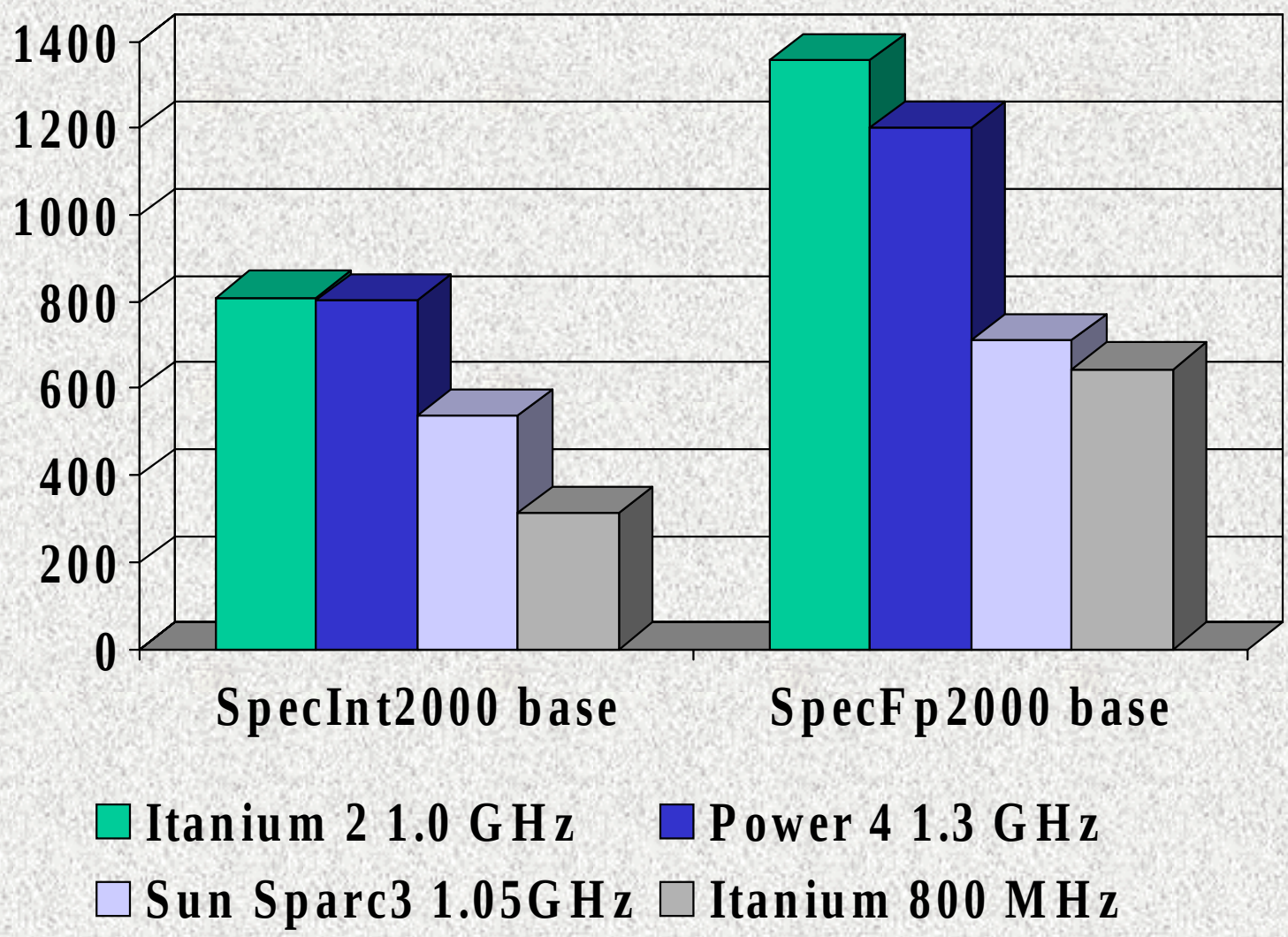
Itanium® 2 Processor Overview

McKinley vital statistics

• Total FETs:	221,000,000
– Core FETs:	40,000,000
– L3 Cache FETs:	181,000,000
• Total FET width:	108 meters
– Core FET width:	39 meters
– L3 Cache FET width:	69 meters
• Total metal route:	70 meters
– Core metal route:	53 meters
– L3 Cache metal route:	17 meters
• C4 bumps:	8088 bumps
– Power C4 bumps:	7789 bumps
– Signal C4 bumps:	229 bumps
• Power:	130 Watts
• Core Frequency:	1 GHz

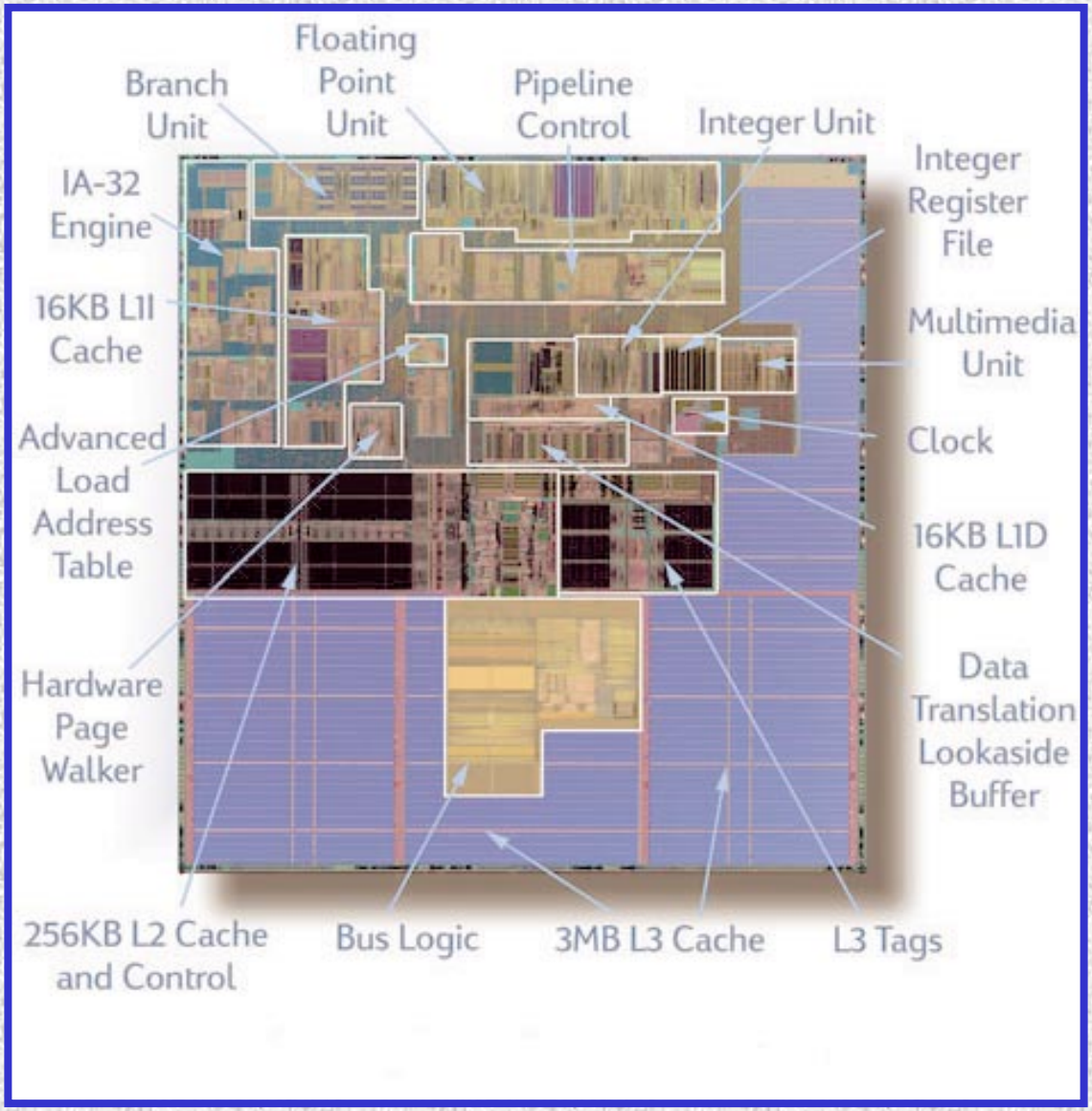
Itanium® 2 Processor Overview

Performance



Itanium® 2 Processor Overview

McKinley Floorplan



Itanium® 2 Processor Overview

- PA-RISC to Itanium translation is straightforward
  - Fixed length RISC instructions are very similar to Itanium instructions
  - 64 bit data
  - PA-RISC compilers have already reordered instructions to increase ILP for current out of order PA-RISC processors
  - HP and others have used binary translation before to get to new architectures
- Largely done through binary translation/emulation as part of the Operating System
  - The OS provides an environment for PA-RISC binaries
  - Binaries are translated on the fly to Itanium instructions
  - The environment dynamically identifies heavily used code sequences
  - If a threshold of usage is reached, “hot” code sequences are translated to native Itanium code for better performance
- Most PA-RISC features are already in Itanium. Very few features added to Itanium to assist PA-RISC emulation
  - addp4 and shladdp4 instructions to assist PA-RISC style address computation
  - PA-RISC integer divide primitive (DS instruction) is absent from Itanium

Itanium® 2 Processor Overview

IA-32 Compatibility

- Itanium requires IA-32 instruction compatibility and a “seamless” interface between Itanium and IA-32. Most common tasks are done in hardware.
- McKinley has microcoded hardware to translate IA-32 instructions:
  - The IA-32 engine replaces the Front End of the pipeline.
  - IA-32 instructions are broken up into strings of equivalent Itanium instructions.
  - The Itanium instructions are injected into the EXP stage of the pipeline.
  - There is additional IA-32 support hardware elsewhere in the pipe for performance and IA-32 specific capabilities (e.g. self modifying code).
- Most of the machine does not distinguish between IA-32 and Itanium modes. Exceptions include: some functional units, interrupt behavior, etc.
- IA-32 pipeline extends beyond the WRB stage for more IA-32 specific support (exceptions, mispredicted branches, etc.)